JAN 1[illegible] 1974

Postel 21255

NIC # 21255
RFC # 607
references: RFC #542

Mark Krilanovich
George Gregg
UCSB

Jan 7, 1974

Comments on the File Transfer Protocol

There are several aspects of the File Transfer Protocol that constitute serious drawbacks. Some of these are quite basic in nature, and imply substantial design changes; these will be discussed in a later RFC. Others could be remedied with very litle effort, and this should be done as soon as possible.

Following is a list of those problems that can be easily solved, together with their proposed solutions:

1. Once a server has been told to be "passive" with regard to establishment of data connections, there is no way for the user to make him "active" again. SOLUTION: define a new command, with a command verb of "ACTV", to mean that the server is to issue a CONNECT rather than a LISTEN on the data socket. If the server is already "active", the command is a no op. "ACTV" is to have the same reply codes as "PASV".

2. Design of an FTP server would be simpler if all command verbs were the same length, and design of an FTP user would be simpler if either all command verbs were the same length, or if multiple blanks were allowed following the verb. SOLUTION: replace the only three-letter verb, "BYE", with a four-letter one, such as "QUIT", and constrain future command verbs to be four letters long.

3. The order of the handshaking elements following a file transfer command is left unspecified. After sending a STOR command, for example, a user process has no way of knowing which to wait for first, the "250 FILE TRANSFER STARTED" reply, or establishment of the data connection. SOLUTION: specify that the server is to send a "250" reply before attempting to establish the data connection. If it is desired to check if the user is logged in, if the file exists, or if the user is to be allowed access to the file, these checks must be made before any reply is sent. The text of the "250" reply would perhaps be more appropriate as "250 OPENING DATA CONNECTION", since it comes before actual data transfer begins. If the server wishes to send an error reply in the event that the data connection cannot be opened, it is to be sent in lieu of the "252 TRANSFER COMPLETE" reply.

4. Some hosts currently send an error reply on receipt of a command that is unimplemented because it is not needed (e.g., "ACCT" or "ALLO"). Even though the text of the reply indicates that the command has been ignored, it is obviously impossible for a user process to know that there is no real "error". SOLUTION: require that any server that does not support a particular command because it is not needed in that system must return a success reply.

5. There is no specified maximum length of a TELNET line, user name, password, account, or pathname. It is true that every system implementing an FTP server likely has different maxima for its own parameters, but it is nearly impossible for the writer of an FTP user (which must converse with many FTP servers) to construct an indefinite length buffer. Typically some arbitrary maximum must be chosen. SOLUTION: specify a maximum length for TELNET lines, user names, passwords, account numbers, and pathnames. This is to be done after conducting a poll of serving sites concerning their individual maxima.

6. The notion of allowing continuation lines to start with arbitrary text solves a minor problem for a few server FTP implementers at the expense of creating a major problem for all user FTP implementers. The logic needed to decode a multi-line reply is unneccessarily complex, and made an order of magnitude more so by the fact that multi-line replies are allowed to be nested. SOLUTION: assign a unique (numeric) reply code, such as "009", to be used on all lines of a multi-line reply after the first.

7. Given that multi-line replies are allowed to be nested, the fact that the maximum allowed level of nesting is left unspecified creates a hardship for implementers of user FTPs. This hardship is somewhat easily solved on a machine that has hardware stacks, but not so for other machines. SOLUTION: specify a maximum level of nesting of multi-line replies.

8. In blocked mode, the protocol states that "all end-of-record markers (EOR) are explicit, including the final one." This prohibits sending data between the final end of record and the end of file, but does not specify what the receiver of data is to do if this rule is broken. That is, should the intervening data be discarded or treated as a new (final) record? SOLUTION: specify that if an end-of-file marker is not immediately preceded by an end-of-record marker, the intervening data is to be discarded.

A major complaint about the protocol concerns the fact that the writer of an FTP user process must handle a considerable number of special cases merely to determine whether or not the last command

sent was successful. It is admitted that the protocol is well-defined in all the following areas, but it is important to realize that the characteristic "well-defined" is necessary, but not sufficient; for many reasons, it is very desirable to employ the simplest mechanism that satisfies all the needs. Following is a list of those drawbacks that unduly complicate the flow chart of an FTP user process:

9. Different commands have different success reply codes. A successful "USER" command, for example, returns a "230", whereas a successful "BYTE" command returns a "200". The concept that success replies should have an even first digit and failure replies an odd first digit does not apply, as "100" means success for "STAT", and "402" means failure for "BYTE". SOLUTION: specify that any command must return a reply code beginning with some unique digit, such as "2", if successful, and anything other than that digit if not successful.

10. Some commands have multiple possible success reply codes, e.g., "USER", "REIN", and "BYE". It is undesirable for an FTP user to be required to keep a list of reply codes for each command, all of which mean "command accepted, continue". SOLUTION: same as for (9.) above. The desire to communicate more specific information than simply "yes" or "no", such as the difficulty in the login procedure that some sites do not need all the parameters, may be solved by having, for example, "236" mean "PASSWORD ACCEPTED, YOU ARE NOW LOGGED IN", and "237" mean "PASSWORD ACCEPTED, ACCOUNT NOW NEEDED". The important point is that the idea of "command accepted" is conveyed by the initial "2", and that finer gradations of meaning can be deduced by the user process, if desired.

11. There are several types of replies that are extraneous from the point of view of a user FTP process, and their reply codes have no characteristic that makes them easily distinguishable. For example, "010 message from operator" and "050 FTP commentary" are superfluous to a user process, and "000 announcing FTP" (in place of "300" greeting) is not. SOLUTION: specify that any reply that has meaning only to a human user and not to a user process must have a reply code beginning with a unique digit, such as "0". The continuation line reply code proposed in (6.) above falls into this catagory, and therefore must start with the same unique digit.

12. The notion of a server sending a "000 announcing FTP" or a "020 expected delay" immediately after completion of the ICP if input cannot be accepted right away is another instance of multiple reply codes having the same meaning to a user process. SOLUTION: require that the server send a reply with a "020" reply

code in the situation cited. If it is desired to communicate more detailed information, the text of the reply may used for this purpose.

In addition to the above mentioned weaknesses in the protocol, the following is believed to be a typographical error:

13. Reply code "331" is cited as a possible success reply code for the commands "BYTE", "SOCK", "PASV", "TYPE", "STRU", "MODE", "ALLO", "REST", "SITE", AND "STAT". This reply code means "ENTER ACCOUNT" (if required as part of login sequence), and perhaps should be "332 LOGIN FIRST, PLEASE". This is especially indicated by the fact that "332" is not listed anywhere in the command-reply correspondance table.

ARPA Network Information Center
Stanford Research Institute
Menlo Park, California 94025

February 1, 1974

Postel

NOTICE OF ERROR

Please note that RFC 607 NIC 21255 "Comments on the File Transfer Protocol" by Mark Krilanovich was sent out to the Network before it was completed.

Therefore, we request that you discard (or at least disregard) this paper. A re-issue will be sent on completion of the document.

Our apologies to Mr. Krilanovich!

RECEIVED
FEB 04 1974